SANDIA REPORT

SAND97–2330 • UC–405
Unlimited Release
Printed September 1997

Massively Parallel Solution of the Inverse Scattering Problem for Integrated Circuit Quality Control

Robert W. Leland, Bruce L. Draper, Sohail Naqvi, Babar Minhas

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Prices available from (615) 576-8401, FTS 626-8401

Available to the public from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd
Springfield, VA 22161

NTIS price codes
Printed copy: A03
Microfiche copy: A01

SAND97–2330
Unlimited Release
Printed September 1997

Distribution
Category UC–405

Massively Parallel Solution of the Inverse Scattering Problem for Integrated Circuit Quality Control

Robert W. Leland[1] and Bruce L. Draper[2]
Sandia National Laboratories
P.O. Box 5800
Alburquerque, NM 87185-0156

Sohail Naqvi and Babar Minhas
Electrical Engineering and Computer Sciences Department
University of New Mexico
Albuquerque, NM 87131

Abstract

We developed and implemented a highly parallel computational algorithm for solution of the inverse scattering problem generated when an integrated circuit is illuminated by laser. The method was used as part of a system to measure diffraction grating line widths on specially fabricated test wafers and the results of the computational analysis were compared with more traditional line-width measurement techniques. We found we were able to measure the line width of singly periodic and doubley periodic diffraction gratings (i.e. 2D and 3D gratings respectively) with accuracy comparable to the best available experimental techniques.

We demonstrated that our parallel code is highly scalable, achieving a scaled parallel efficiency of 90% or more on typical problems running on 1024 processors. We also made substantial improvements to the algorithmics and our original implementation of Rigorous Coupled Waveform Analysis, the underlying computational technique. These resulted in computational speed-ups of two orders of magnitude in some test problems. By combining these algorithmic improvements with parallelism we achieve speedups of between a few thousand and hundreds of thousands over the original engineering code. This made the laser diffraction measurement technique practical.

[1] Parallel Computing Sciences Department, Mailstop 0441, (505) 845-8439, leland@sandia.gov.
[2] Advanced Radiation Hardened CMOS Technologies Department, Mailstop 1074, (505) 844-7865, draperbl@sandia.gov.

1. Background. When an integrated circuit is illuminated by laser, an interference pattern is created in the diffracted light. Geometric properties of the chip surface can, in principle, be inferred from computational analysis of the scattered radiation; this information can then be used to monitor and improve process quality. In most conventional methods of monitoring the chip patterning processes, wafers are removed from the line following key steps and inspected using standard white light microscopy – a technique that is straightforward but which yields only one-dimensional information. Other more rigorous inspection/measurement techniques used in the IC industry (e.g. scanning electron microscopy) can give 3D information but are very time consuming and destructive. With the new laser diffraction method described in this report, a chip can be tested rapidly, automatically and non-destructively at many stages of processing. When mature, this technology is expected to decrease costs, speed process flow and increase yield, and hence be of substantial economic significance.

Prior to the work described in this report, The University of New Mexico (UNM) had developed a prototype of the necessary laser scattering hardware and had used this successfully to make 1D measurements on chips etched with simple diffraction gratings [1, 6]. This work, however, relied for solution of the inverse problem on an analytical result which does not hold for measurement of 2D or 3D samples. In the higher dimensions the only known method of solving the inverse scattering problem was and is to solve the forward problem repeatedly and use each newly calculated field to improve the previous approximation to the true geometry.

Simple estimates showed that this iterative method would clearly require supercomputing capacity, and Sandia's experience at the Massively Parallel Computing Research Laboratory (MPCRL) indicated that use of a parallel supercomputer would likely result in a faster solution at much lower cost than use of a traditional vector supercomputer. Hence there was a natural way to cooperate on the solution.

2. Project Summary. When we began, we were primarily interested in showing that the inverse problem was tractable by any means. We believed that eventually we could produce a highly efficient parallel algorithm that would run on a supercomputer stationed at the fabrication facility. Accordingly, we were aiming at a "proof of concept" result, and our goals were to:

- Devise a highly parallel solution of the inverse scattering problem and implement it on Sandia's 1024 node Ncube2 parallel supercomputer.
- Confirm the accuracy of the technique by analyzing an array of samples fabricated in Sandia's Microelectronic Development Laboratory (MDL).
- Establish that the algorithm's performance scaled with the number of processors used and hence demonstrate the capability to solve realistic problems given sufficient resources.

To accomplish these goals we envisioned the following: A serial code developed at the University of New Mexico for the 2D forward scattering problem would be restructured to handle the 3D problem and run on the Ncube2. The code's direct linear

solver would then be replaced with an asymmetric Krylov solver employing polynomial preconditioning since these rely on repeated matrix-vector multiplication, an operation which parallelizes well. The code would then be tuned until it was fast enough to serve as the kernel in an iterative method for solving the inverse problem. The iterative algorithm for solving the inverse problem would be developed jointly with electromagnetics experts at UNM while appropriate 3D test samples were designed and fabricated in Department 2131 at Sandia. Data would be collected on the chip samples using the interferometry apparatus at UNM and fed into the full code. The code predictions would then be checked against measurements taken by conventional means. Once the code was verified, its scalability would be established using the modeling techniques developed in [3].

As the work progressed, we realized there was a more practical paradigm we could follow. In the procedure described above, each inverse problem is solved independently of previous work by a presumably relatively small number of functional evaluations. This is the appropriate approach when the number of mask designs to be considered is large, and there are relatively few of each design produced. This fit the production pattern at the MDL where production is highly customized and lot sizes are very small.

But in a more typical industrial setting, one mask is used for very high volume production. In that setting, each inverse problem is solved using a small number of function evaluations also, but these function evaluations are computed in a relatively restricted parameter space corresponding to the known or expected variations experienced with a given mask or fabrication technology. This means in practice that the effective domain of the scattering solutions is known a-priori, and also that there is significant overlap in the scattering solutions required when computing a large number of inverse solutions. The right strategy, therefore, in a high volume production setting is to pre-compute this space of scattering solutions and use a table look-up in the inverse solution phase. This approach made the parallel computing aspect of the problem much easier as we could, once we did a little work to fit the calculations corresponding to one point in the parameter space on a single processor, simply divide up the parameter space across the processors in order to compute the look-up table. Hence emphasis shifted to whether we could go a step further then originally envisioned and demonstrate a prototype system that would not require the fabrication facility to maintain a supercomputing capability.

In subsequent discussion, we developed the following conceptual design for such a system: A Computer Aided Design (CAD) mask for a new chip containing an appropriate test pattern would be submitted to a large multiprocessing system (at, e.g., the MPCRL) and a concise list of a few hundred neural network training parameters returned. These parameters would be fed into a small computer attached to a laser interferometry apparatus which fired at each chip as it passed down the production line. Chips that did not meet specification would be detected and rejected in real-time by the small computer running a neural network pattern matching algorithm. Appendix 1 shows a schematic of the system corresponding to this conceptual design.

At the end of the first year, we had passed the following milestones in development of this system:

- Designed in the MDL a mask for the special test chip required.
- Adapted the code solving the 2D forward electromagnetic scattering code to run in parallel on the Ncube2 hypercube multiprocessor with high parallel efficiency and nearly perfect scalability.
- Derived the appropriate 3D electromagnetic scattering equations (a significant extension from the 2D case).
- Implemented a pattern matching neural network which used the output of the 2D forward scattering code to solve the inverse scattering problem.
- Compared the output of the neural network with that of a classical pattern matching technique (linear least squares) and experimental results (from UNM's apparatus) and found excellent agreement.

Hence we deviated, as discussed previously, from the original plan in two important ways:

- We used a more straightforward parallelism than we originally planned in adapting the 2D forward scattering code to the Ncube. This was more efficient and natural in the 2D case, but it was not clear whether a similar simplification would be possible in the 3D case.
- We decided to investigate using a Multilayer Perceptron neural network to implement the pattern matching algorithm. This approach was not mentioned in the original proposal, and it was not clear that its use in solving the inverse problem was justified. On simple problems the classical linear least squares approach actually worked better. But the neural network approach offered an attractive flexibility, and we felt that on more difficult or higher dimensional geometries, it might also work better.

In the second year, we accomplished the following:

- Fabricated a test chip from mask.
- Implemented the 3D forward electromagnetic scattering equations in serial.
- Converted the 3D serial code to run in parallel on the Ncube2.
- Used the 3D forward scattering data to train and test the neural network designed to solve the inverse problem.
- Compared the neural network and classical results pattern matching algorithm test results.
- Verified the pattern matching code on the test chip.

The major developments here were:

- We determined that the classical pattern matching technique was preferable to the MLP neural network approach since it achieved comparable results more simply and efficiently.
- We showed that we were able to measure diffraction grating parameters with accuracy comparable to the best previously existing methods (which are less desirable because they are either invasive or require human intervention and

hence are disruptive of production flow).

3. Technical Detail.

3.1. Improvements in RCWA Technique. The theory underlying our computational work is known as Rigorous Coupled Wave Analysis (RCWA) and is best laid out by Moharam and Gaylord [5]. We describe several improvements and extensions assuming their development and notation.

The RCWA technique relies crucially on the computation of eigenvalues of a structured matrix and the solution of an associated linear system. Most of the compute time is spent in either the eigensolve on the matrix A or the linear solve on the matrix Q. Hence we tried to reduce the time spent in these kernels. We found four ways to do this:

Method 1: One method is based on the observation that not all modes must be retained in the analysis. A plot of the minimum absolute value and minimum imaginary value of the eigenvalue for each mode number m (where A is of dimension $2m$) shows that these quantities become nearly constant after a certain mode number, suggesting that the eigenvalues having the lower absolute value dominate system behavior.

In order to verify this, we ran the forward scattering program for sufficiently large number of modes and constructed the Q matrix starting from the lowest eigenvalues. We plotted the power diffracted in the various propagating modes and noted that this tended toward a constant. The power diffracted in the various modes is nearly the same in both cases after a certain mode number, supporting the conjecture that the lower absolute value eigenvalues dominate. Hence we could solve for the eigenvalues just once and iterate the linearsolve subroutine until the power diffracted in the propagating modes became constant.

A simple complexity analysis shows that this method is indeed faster than the previous method. In the original algorithm, the number of modes to retain, m, is selected, then the A matrix is constructed. The eigenvectors of A are computed, requiring to first order $40m^3$ work. The eigenvectors are used to construct the linear system with matrix operator Q. This system is solved, requiring to first order $64M^3$ work. If the diffracted power has not converged, m is increase by 2 and the process is repeated. In the modified approach, the eigenvectors are computed just once. Suppose it takes p iterations to obtain convergence in the power diffracted in the modes. Then to first order the time taken in this portion of the original program is proportional to $104\sum_{i=0}^{i=p-1}(m+2i)^3$. With the modified approach, the time is proportional to $40m^3 + 64\sum_{i=0}^{i=p-1}(m+2i)^3$. Hence the ratio R of the time of the modified case over the original is

$$R = .615 + .385\frac{m^3}{\sum_{i=0}^{i=p-1}(m+2i)^3} \tag{1}$$

hence the new method may be as much as 1.6 times faster than the old.

Method 2: The second way to reduce the compute time is to effectively reduce the dimensionality of the eigenproblem to be solved. Consider the second order differential equation implemented by the forward scattering program using the state-variable

method as described in [5]:

$$d^2S_i(z)/dz^2 - j2k_{x0}dS_i(z)/dz = (k_{xi}^2 + k_{x0}^2)S_i(z) - k^2\sum_p \epsilon_p(z)S_{i-p}(z) \tag{2}$$

Defining the state-variables as $S_{1,i}(Z) = S_i(Z)$ and $S_{2,i} = dS_i(Z)/dz$ transforms this set of second order differential equations into two sets of first order differential equations. Implementing this in matrix form, the system can be written $\dot{\underline{S}} = A\underline{S}$, where A is the system matrix whose eigenvalues we determine. This matrix has a definite structure independent of dimension. For example, for $m = 3$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ d1 & a1 & a2 & b & 0 & 0 \\ a1 & d2 & a1 & 0 & b & 0 \\ a2 & a1 & d3 & 0 & 0 & b \end{bmatrix} \tag{3}$$

where $b = j2k_{z0}$.

We have to solve the system $(A - \lambda I)\vec{x} = 0$, where

$$(A - \lambda I) = \begin{bmatrix} -\lambda & 0 & 0 & 1 & 0 & 0 \\ 0 & -\lambda & 0 & 0 & 1 & 0 \\ 0 & 0 & -\lambda & 0 & 0 & 1 \\ d1 & a1 & a2 & b-\lambda & 0 & 0 \\ a1 & d2 & a1 & 0 & b-\lambda & 0 \\ a2 & a1 & d3 & 0 & 0 & b-\lambda \end{bmatrix} = \begin{bmatrix} \mathbf{\Lambda} & \mathbf{I} \\ \mathbf{\Upsilon} & \mathbf{B} \end{bmatrix} \tag{4}$$

and $\mathbf{\Lambda}, \mathbf{I}, \mathbf{\Upsilon}$ and $\mathbf{B}$ are all, more generally, of dimension $m \times m$. An equivalent system is

$$\begin{bmatrix} \mathbf{\Upsilon} & \mathbf{B} \\ \mathbf{\Lambda} & \mathbf{I} \end{bmatrix} \vec{x} = 0. \tag{5}$$

Multiplying on the left by

$$\begin{bmatrix} \mathbf{I} & -\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{6}$$

we arrive at the system

$$\begin{bmatrix} \mathbf{\Upsilon} - \mathbf{B\Lambda I} & \mathbf{0} \\ \mathbf{\Lambda} & \mathbf{I} \end{bmatrix} \vec{x} = 0. \tag{7}$$

The eigenvalues can thus be determined by evaluating, $|\mathbf{\Lambda B} - \mathbf{\Upsilon}| = 0$ where $\forall i \in (1, m)$, $\mathbf{\Lambda B}_{i,i} = -\lambda(j2k_{zo} - \lambda) = \lambda^2 - \lambda j2k_{z0}$. Therefore we need only find the eigenvalues of the $\mathbf{\Upsilon}$ matrix and then compute the eigenvalues of the full matrix,

A, using the quadratic formula $\lambda^2 - \lambda j 2k_{zo} - \alpha_l = 0$ where α_l is the l_{th} eigenvalue of the Υ matrix, *i.e.*

$$\lambda = jk_{z0} \pm \sqrt{-k_{z0}^2 + \alpha_l}. \tag{8}$$

Hence we have effectively halved the dimensionality of the eigenproblem to be solved. Since this is a cubic operation, this saves us about a factor of 8 in work associated with solving the eigenproblem.

Method 3: A third way is to select the value of m which would give convergence in the diffracted power with minimal iteration over the linear solve. We start by trying to develop a physical interpretation of our representation of the electric field in the grating region:

$$E = \sum_i S_i(z) \exp[-j(k_{zi}x + k_{z0}z)] \tag{9}$$

On substitution of (9) in the wave equation we get (2), whose eigenvalues we evaluate. Thus the i^{th} space harmonic is given by

$$S_i(z) = \sum_{-l}^{l} C_l \chi_{il} \exp(\lambda_l z) \tag{10}$$

with $l \neq 0$. If we substitute (10) and (8) in (9) we can write the electric field in the grating region

$$E = \sum_i \sum_{-l}^{l} C_l \chi_{il} \exp[\pm z\sqrt{-k_{z0}^2 + \alpha_l}] \exp[-j(k_{xi}x)] \tag{11}$$

again with $l \neq 0$. Since the field is represented as sum of infinite number of modes, for any i_{th} mode we can write

$$E_i = \sum_{-l}^{l} C_l \chi_{il} \exp[\pm \zeta_l z \exp[-j(k_{xi}x)] \tag{12}$$

where $\zeta_l = \sqrt{-k_{zo}^2 + \alpha_l})$. Now consider the following grating cases.

Lossless grating: In this case the eigenvalues of the Υ matrix are always real, and they always increase monotonically, starting from a negative value. So ζ will be initially imaginary and will become real as m increases. Therefore the representation of each mode will contain finite imaginary exponential factors and infinite real exponential factors. We conjecture that convergence starts when all the imaginary exponential factors are included in the sum, so the minimum value of m will be that which makes ζ real. This conjecture is verifiable by plotting the data from the forward scattering program. Using this method we find the minimum mode number to start from and we rapidly get convergence in the diffracted power a few modes above this mode number.

Lossy grating: The eigenvalues of the Υ matrix are complex in this case, and so ζ will be complex. A plot of the imaginary part of ζ versus mode number (for two

different values of the complex index of refraction) indicates that the imaginary part decreases monotonically. So we can say as before that convergence will start when we have included all the imaginary exponential factors (*i.e.* when ζ becomes purely real). But in this case ζ converges very slowly to a real value, so in practice it is feasible to use that mode number which makes the imaginary pat of ζ arbitrarily small. We chose the number of modes required for the imaginary part of ζ to become 0.1 as a function of imaginary the part of the grating index. Using this criterion we can again find the value of m where convergence starts. In our experiments we found that for metallic gratings a very large number of modes were required.

It is difficult to quantify the savings associated with this technique because they are very data dependent. But a simple characterization can be arrived at starting with the complexity expressions developed in connection with method 1. The terms are all proportional to (the final value of) m^3. If the technique outlined here permits us to stop β modes short of where we otherwise would have, the terms will then be proportional to $(m-\beta)^3$. Hence the ratio T of work in the new method to the old will be

$$T \approx 1 - 3(\frac{\beta}{m}) \tag{13}$$

if $\beta << m$ (which is conservative since if β is comparable to m the savings are greater). So, for example, if $\beta/m = .1$, we save 29%.

Method 4: The last source of improvement is not so much based on mathematics as computer logic. When the height parameter is increased an incremental amount, most of the calculation is the same as for the previous height. Hence by saving the basis functions computed on the previous slice, we avoid most of the calculation. Since we would have to compute on all slices once anyway for the greatest height examined, this observation effectively removes the height dimension from the parameter space. Put another way, we get this variation almost for free if we do the calculation in the correct order. Since we might typically use 10 to 100 or more slices in a calculation, this savings can be very substantial if significant variation in height is a goal of the parameter study - we may save as much as two orders of magnitude in one of the most costly portions of the code over the original approach.

3.2. Extension of RCWA to Three Dimensions. In addition to these improvements to the singly periodic case (*i.e.* the 2D case), we derived the RCWA equations for the doubly periodic (3D) case. This derivation is presented in Appendix 2. Note that in this development the terminology "general two dimensional case" is still used to refer to what would generally be described as the three dimensional problem because the focus is on the extension of the "singly periodic" diffraction patterns consisting of lines in the plane to the "doubly periodic" patterns shown.

3.3. Parallel Implementation. As indicated in the introductory text, the parallel implementation of the scattering is actually fairly straight forward. The code is written in Fortran 77 with some calls to routines for message passing provided by the machine vendors (Ncube or Intel SSC). A functional description of the main routine `prog.f` follows. Complete source code is available from the first listed author.

After setting some initial parameters that control buffer sizes, the code sets up several common blocks containing essential information used in many routines, e.g. the type of machine the code is intended to run on (Sun workstation, the Ncube2 or the Intel Paragon). Then the machine configuration is ascertained so that each processor will know the number or processors in use, its own processor identification number, the maximum processor id in use and its position in the pipeline discussed subsequently. Processor 0 then opens the files for input and output.

All processors then call the routine `get_paramset` in order to obtain their portion of the parameter space. This requires processor 0 to read the parameters from the input file `fin` (this and similar names can be changed in `prog.f`). Processor 0 packs these parameters into a vector of minimal length which is broadcast to all other processors using a standard binary fanout algorithm which is particularly good for the Ncube2's hypercube architecture, but which also works well on the Intel's mesh architecture. (Note however that the use of this algorithm does restrict the current implementation to use a power of two number of processors.) The other processors receive the parameter vector and unpack it into the correct parameter variable locations. These parameters are then echoed to the screen and the output file `fout` by processor 0.

Next timing registers are set up, and processor 0 steps through the total parameter range without doing the scattering calculations to count and echo the number of variations in each parameter. The echoing serves as a useful check since it is very easy to specify a parameter range other than the one intended. With it switched on, there is no ambiguity about whether endpoints are included in the range, etc. The exact variation to be computed over is explicitly calculated and displayed. The count is used in mapping the parameter sets to processors and also in checking whether enough buffer memory was allocated earlier. If not, the program exits with instructions to change parameters and recompile. If memory could be conserved in this problem by recompiling with smaller parameters, this is also noted.

In the next phase, a variety of electromagnetic parameters are computed and the main loops over the input parameters are set up. Each processor then picks out a range of the parameter space based on its position as assigned in `get_config`. The electromagnetic computations described in [5] as modified in 3.1 are performed by each processor with one set of parameters from within its range of the parameter space.

In the final phase of the computation, the diffracted powers are packed into a vector and these are sent to processor zero in pipeline fashion. Processor 0 writes one result to either the screen or `fout` or both (or neither) depending on how the io parameters are set. The pipelining allows processor 0 to read the next result to be written while it is writing the current one, thereby increasing efficiency over a polling approach. If any processor has parameter sets yet to compute on, the process is repeated until these are exhausted. Finally timing information collected on processor 0 (which will certainly be the last to finish given its io responsibilities) is written to the screen or `fout`.

3.4. Using the Parallel Code. The code is packaged for distribution along with appropriate makefiles and sample input and output files. This section describes use of the code in that context

As indicated earlier, the appropriate material, optical and geometric properties are specified in an input file, `fin`. The ones most commonly varied are the parameters which describe the geometry of the diffraction grating. The lines of the diffraction grating are described by their width, depth (or, interchangeably, their "height") and the angle of the sidewalls (left-right symmetry is assumed). Another important geometric parameter is the incidence angle of the laser. These four parameters are those typically varied in the generation of the table of forward scattering patterns. The sample input file `fin.small` highlights these by capitalizing the parameter names in the comments. On the same line the range of each parameter is specified by a start, end and increment value (in that order). The sample input file `fin.big` has reasonable parameter ranges which generates 108 distinct scattering computations.

As described earlier, the resolution of the computation and output is controlled by parameter values set in the main routine contained in file `prog.f`. In particular, the parameter `n` (set on line 36) controls the number of diffracted power orders (different angles) at which the power is computed. The run time is cubic in this parameter. A value of 10 is a reasonable starting point for experimentation. The power spectra will have $2n+1$ entries, though often many or most of these will be zero. If you don't want to have all of them printed, change the `nsideorders` parameter on line 120 (By convention each sideorder has a positive and negative instantiation, so the number actually printed is $2 * \texttt{nsideorders} + 1$.)

The output level is controlled by the flag `outmode` set on line 56. As the comments above it indicate, increasing numbers mean more detailed output. For big runs you probably want to set outmode to 1. Lastly, where the output is printed is controlled by two more flags, `toscr` set on line 57 and `tofout` set on line 58. If `toscr` is `TRUE` the output is written to the screen. If `tofout` is `TRUE` most of the output (except for some unimportant echoing of the parameter ranges) is written to the `fout`.

To compile on the sun you want to execute the command: `make -f makefile.sun` The executable is written to the file `prog`, so to run you just type `prog`.

You can also compile for the Ncube2 with `makefile.ncube` or the Intel Paragon with `makefile.intel`. You just have to change the machine flag on line 155 of `prog.f` if you want to use the Ncube or Intel. To execute you then need to apply the `xnc` command on the Ncube or the `yod` command on the Intel to the executable.

4. Results.

4.1. Simulation Results. Figure (1) shows a comparison of predicted and measured diffraction grating linewidth data. Alphastep is a standard measuring technique that drags a stylus across the grating, destroying it in the process. The Chemometrics approach uses the forward scattering code here in conjunction with a linear least squares technique to solve the inverse problem. The neural network approach also uses the forward scattering code and a neural network pattern matching code developed in conjunction with this project to solve the inverse problem. Note the excellent agreement between the measured linewidth of 0.90 microns and the predicted linewidth (by either technique) of 0.88 microns.

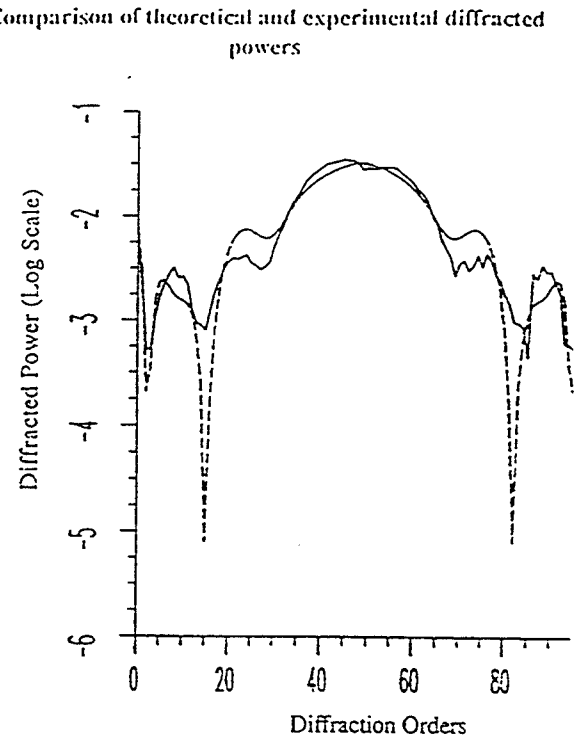


Fig. 1. Comparison of measured linewidth (solid line) and linewidth predicted by the computational technique of this report (dotted line). Excellent agreement was found between measured and predicted linewidths.

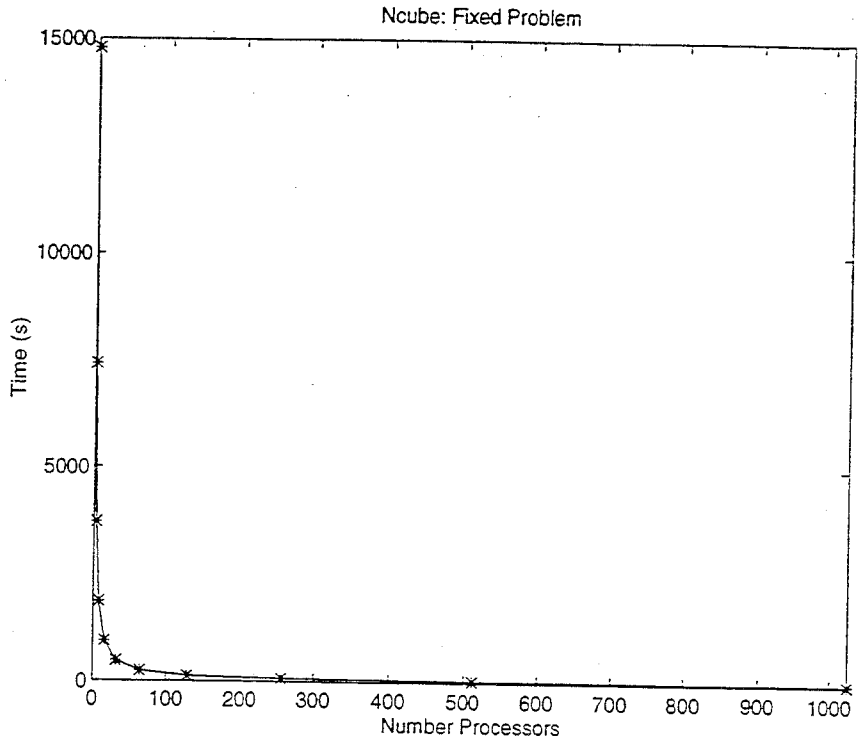


Fig. 2. Time to complete a fixed problem as a function of number of Ncube2 processors applied.

4.2. Parallel Performance. Figure (2) shows the time to complete a relatively large, fixed problem as a function of the number of Ncube processors applied. This illustrates expected, classical behavior. The execution time is driven down significantly, but the efficiency of the computation degrades as the number of processors applied increases as shown in fig. (3) (note the horizontal axis is now hypercube dimension, *i.e.* $\log_2(P)$ where P is the number of processors since this serves to clarify the plot. This degradation is directly attributable to the increase in io overhead as a fraction of the total time spent since that is the entire serial content of the program; see fig. (4). By plotting the log of the execution time against hypercube dimension we can see from the slope of the curve that the program retains good scalability until about 256 processors at which point efficiency begins to degrade more noticeably.

Figure (6) shows performance on a problem whose size scales with the number of processors applied. So each processor always does (in this case) one parameter set. Notice here that execution time stays very nearly constant from 1 to 1024 processors. This again is the expected behavior for a well designed code in which the serial work stays in proportion to the work that can be done in parallel. We see from fig. (7) that this translates into high "scaled" efficiency across the entire range of hypercubes available. There is a slight degradation in efficiency because, as fig. (8) shows, the io overhead does increase. This is primarily because with more parameter sets, there are more results to write back to a file. Nevertheless, 90% efficiency on this size problem (order a minute execution time) at 1024 processors is very good. We could easily make that near unit effiency by running problems near the memory capacity of the individual processors. Then nearly the same io overhead could be amortized over run times of hours in length.

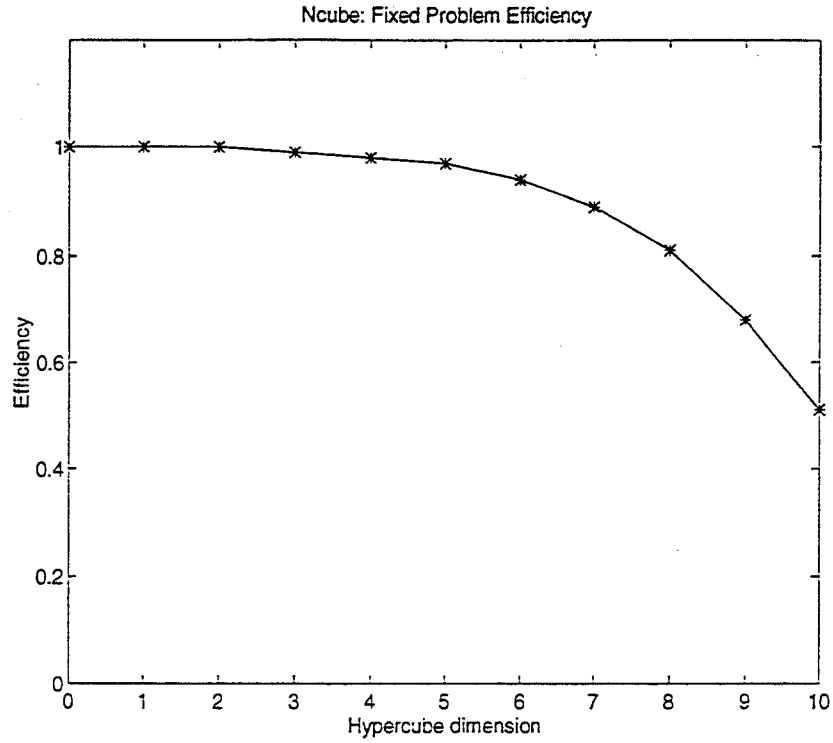


Fig. 3. Efficiency on a fixed problem as a function of Ncube2 hypercube dimension.

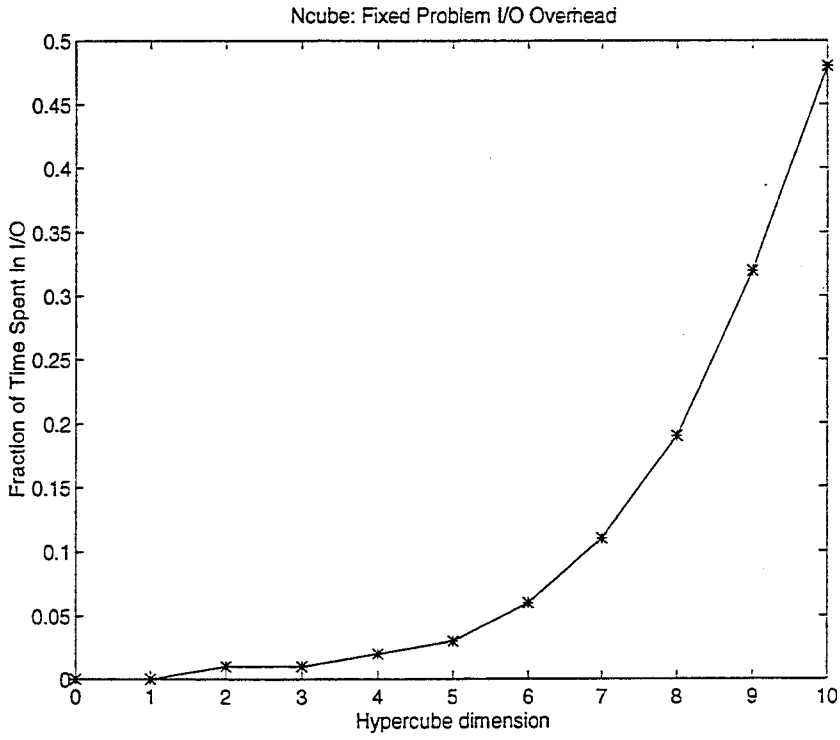


Fig. 4. Input/output overhead as a function of Ncube2 hypercube dimension.

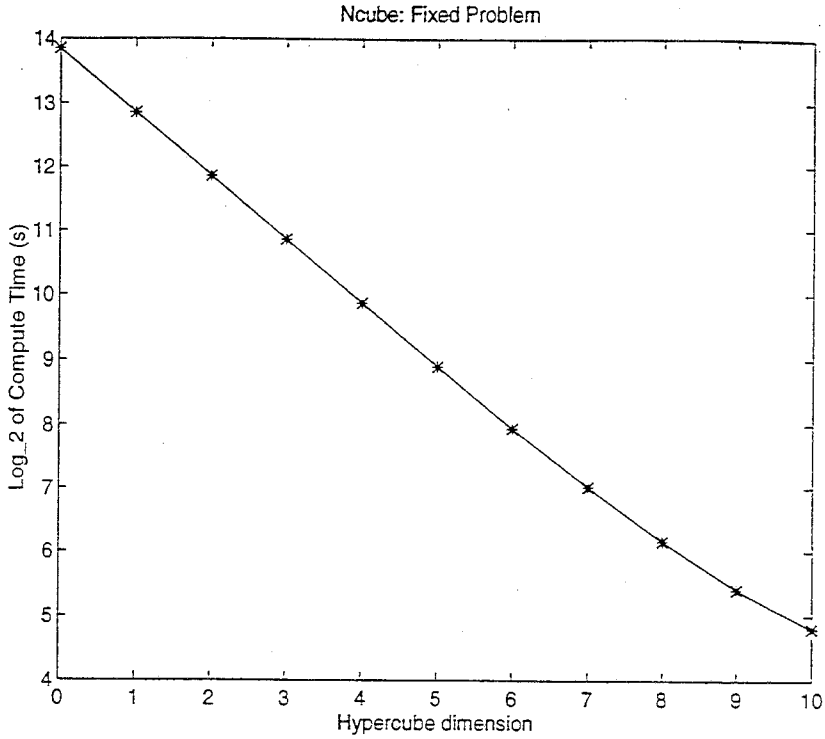


Fig. 5. Log/log plot of total execution time versus number of Ncube2 processors applied showing performance leaving the linear trajectory significantly at about 256 processors.

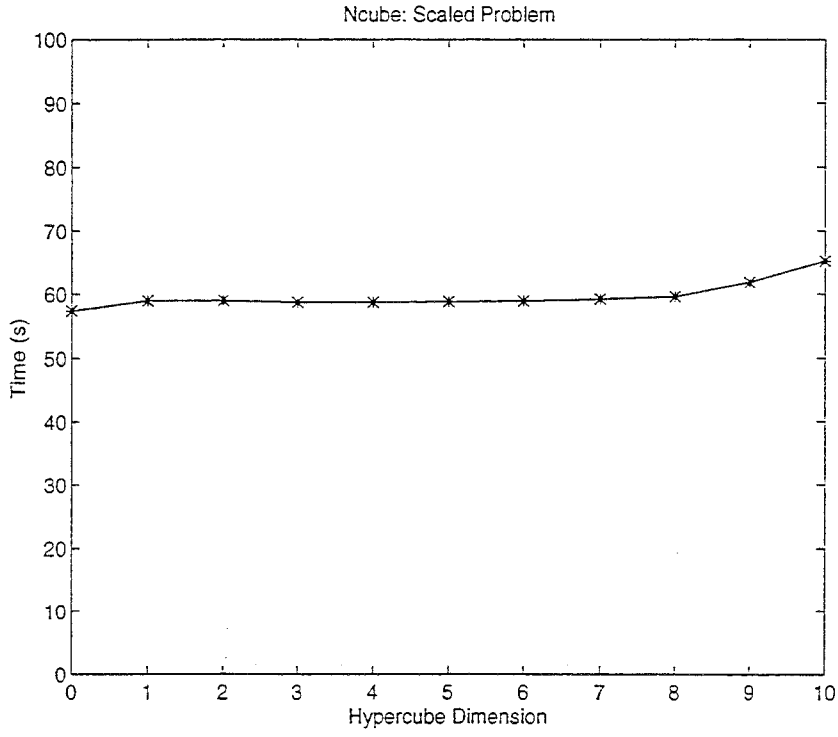


Fig. 6. Execution time for a scaled problem as a function of hypercube dimension.

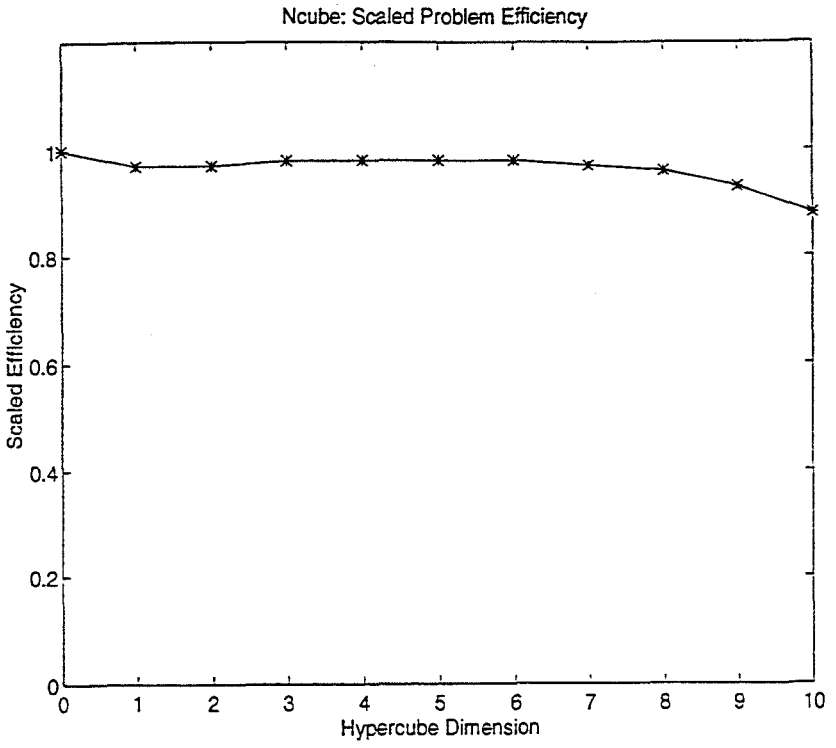


Fig. 7. Efficiency for a scaled problem as a function of hypercube dimension.

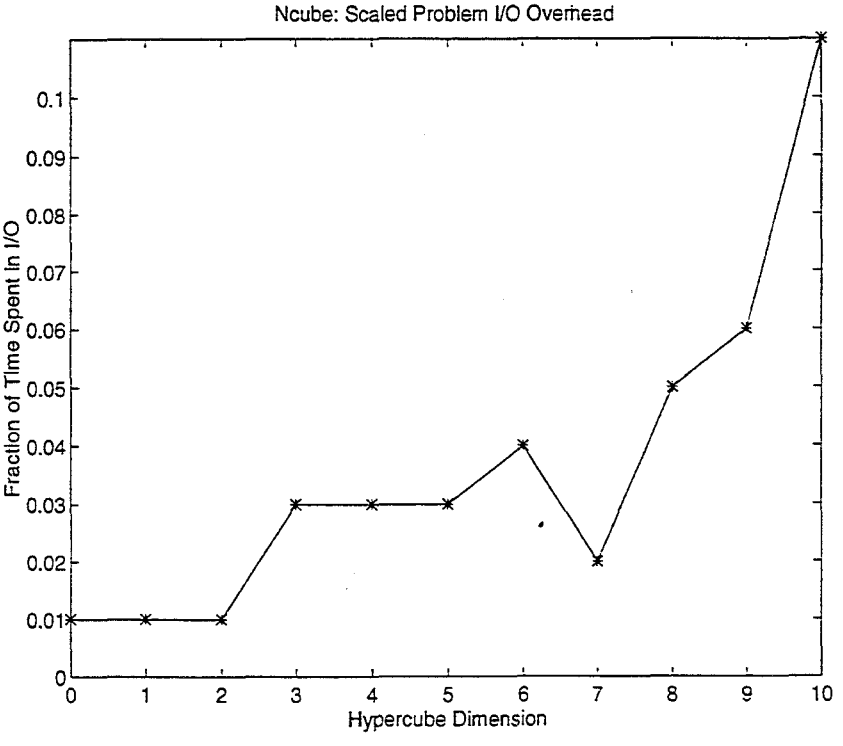


Fig. 8. Input/output overhead for a scaled problem as a function of hypercube dimension.

5. Conclusions. We were able to dramatically reduce the compute time needed to check the quality of test patterns on integrated circuits using laser diffraction. A summary of the various improvements in the computational technique is shown in table (1).

Improvement	Expression	Typical Net Impact
Method 1	R^{-1}	1.5
Method 2	$8 \times$ `time_eig_solve`	1.5
Method 3	T	1.4
Method 4	`nslices`	50
Parallelism	`par_eff` $\times$ `nprocs`	1000
Net effect		157,000

Table 1. Speedups resulting from the various improvements to the original engineering code.

As a result of this project, a number of publications were generated which have advanced work in the field of laser diffraction: [8, 7, 4, 9, 2]. Further detail on the work can be found here and in the previously cited works.

REFERENCES

[1] K. C. HICKMAN, S. S. GASPAR, S. S. NAQVI, K. P. BISHOP, J. R. MCNEIL, G. D. TIPTON, B. L. DRAPER, AND B. R. STALLARD, *Use of diffraction from latent images to improve lithography control*, Integrated Circuit Metrology, Inspection and Process Control V, SPIE 1464 (1991).

[2] R. KRUKAR, R. W. LELAND, AND S. R. MCNEIL, *Determining the limit for diffraction based metrology*, in Machine Vision Applications in Industrial Inspections II, SPIE Symposium on Electronic Imaging Science and Technology, February 1994.

[3] R. W. LELAND, *The Effectiveness of Parallel Iterative Algorithms for Solution of Large Sparse Linear Systems*, PhD thesis, University of Oxford, Oxford University Computing Laboratory, Oxford, England, October 1989.

[4] B. MINHAS, S. S. NAQVI, AND R. W. LELAND, *A study of eigenvalue behavior in rigorous coupled wave analysis*, in Optical Society of America Annual Meeting, October 1993.

[5] M. G. MOHARAM AND T. K. GAYLORD, *Diffraction analysis of dielectric surface-relief gratings*, Journal of the Optical Society of America, 72 (1982), pp. 1385–1392.

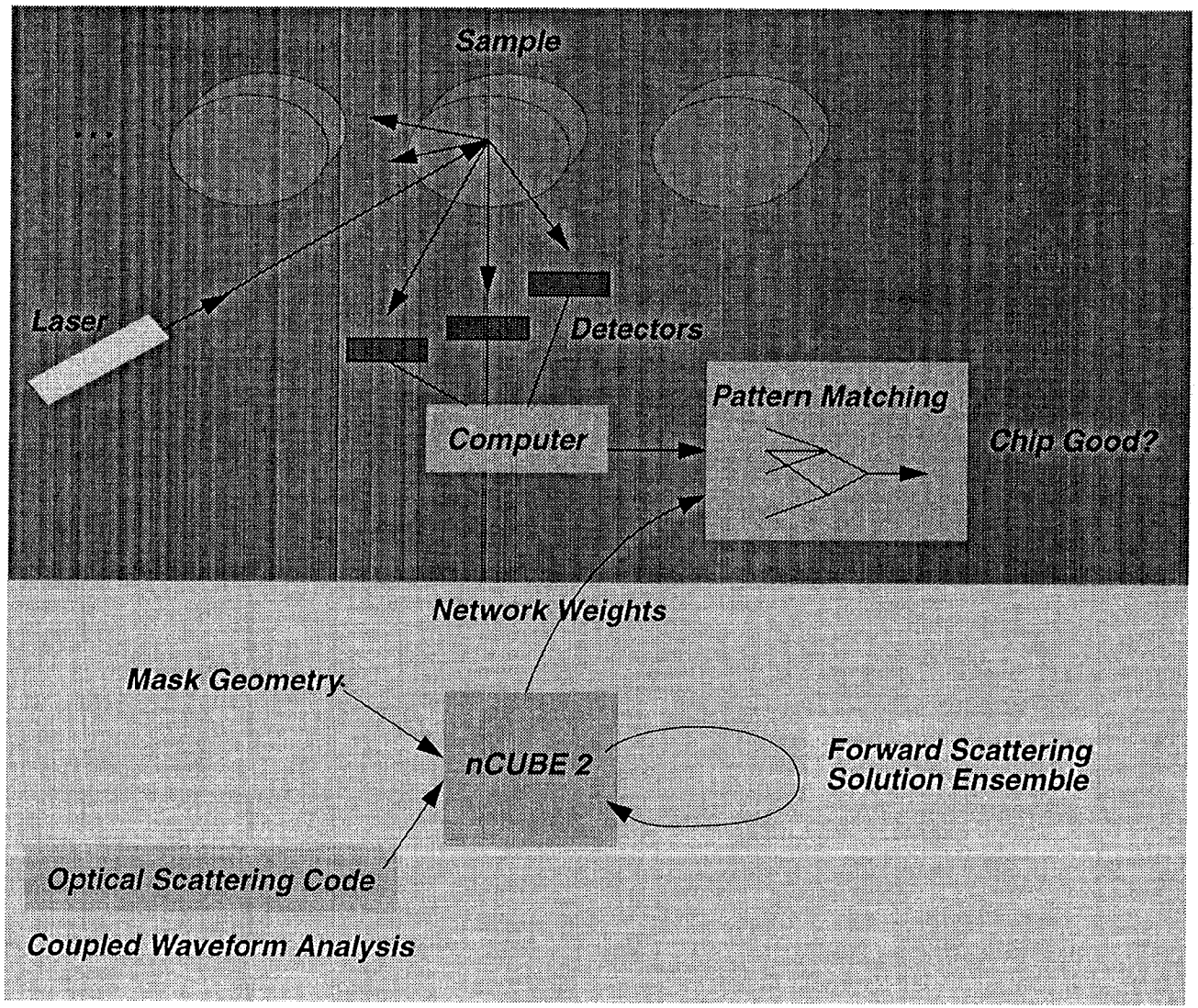
[6] S. S. NAQVI, S. M. GASPAR, K. C. HICKMAN, AND J. R. MCNEIL, *A simple technique for linewidth measurement of gratings on photomasks*, Integrated Circuit Metrology, Inspection and Process Control IV, SPIE 1261 (1990), pp. 495–504.

[7] S. S. NAQVI AND R. W. LELAND, *Massively parallel solution of scattering from doubly periodic structures*, in Optical Society of America Annual Meeting, 1992.

[8] S. S. NAQVI, J. R. MCNEIL, R. KRUKAR, FRANCO, R. W. LELAND, AND K. P. BISHOP, *First principle simulation of diffraction based metrology techniques*, Microelectronics World, (1993).

[9] S. S. NAQVI, B. MINHAS, R. KRUKAR, AND R. W. LELAND, *Electromagnetic scattering analysis of diffraction gratings*, in Optical Society of America Annual Meeting, October 1993.

Schematic



3D algorithm

↓

3D parallel code

↓

Trainer

↓

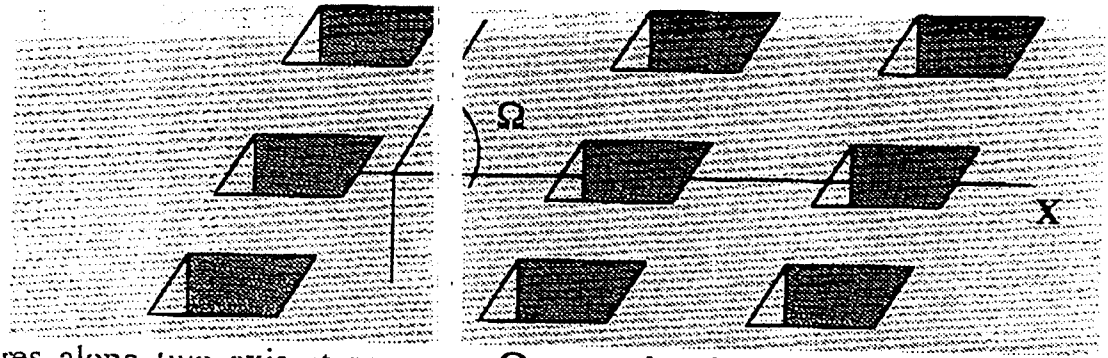
Lab Demo

- Patterns on chips
- Identify actual variabilities

Appendix 2

Introduction

The procedure developed for the sol ion of line gratings using coupled wave analysis can be conveniently extended to be appl able in the general two dimensional case. The scattering surface is, given in the figure ove and is assumed to be infinite dimensional



with apertures along two axis at an an e Ω to each other. Similar to the arbitrary polarization case the E_x and E_y compor nts of the electric field can not be considered separately in general and both components ave to be considered simultaneously to obtain a solution for the scattered field. The incide field is assumed to be an arbitrarily polarized plane wave obliquely incident on the scat rer placed in the z = 0 plane.

Problem Formulation

A general representation of the total ctric field in regions 1 and 3 can be written as,

$$\begin{bmatrix} E_{1x} \\ E_{1y} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} Exp(-j\vec{k}_1.\vec{r}) + \sum_{m=-\infty}^{\infty} \begin{bmatrix} R_x \\ R_y \end{bmatrix} Exp(-j\vec{k}_{1m}.\hat{z}z) f_{1m}(x,y).$$

Where,

$$f_m(x,y) = Exp(-j(\vec{k}_{1m}.\hat{x}x + \vec{k}_1.\hat{y}y)).$$

$$\vec{k}_{1m}.\hat{x} = \vec{k}_1.\hat{x} - \frac{2\pi m}{d},$$

$$\vec{k}_1 m.\hat{z} = \begin{cases} -\left[k_1^2 - [(\vec{k}_{1m}.\hat{x})^2 + (\vec{k}_1.\hat{y}\ |\right]^{1/2}, & case(a), \\ j\left[(\vec{k}_{1m}.\hat{x})^2 + (\vec{k}_1.\hat{y})^2 - k^{\prime}\ \right]^{1/2}, & case(b). \end{cases}$$

Where, $case(a) = k_1^2 \geq [(\vec{k}_{1m}\ )^2 + (\vec{k}_1.\hat{y})^2]^{1/2},$

$case(b) = [(\vec{k}_{1m}.\hat{x})^2 + (\vec{k}_1.\hat{y})^2]^{1/2} \geq {}_1{}^2.$

For the transmitted fields, we have that

$$\begin{bmatrix} E_{3x} \\ E_{3y} \end{bmatrix} = \sum_{m=-\infty}^{\infty} \begin{bmatrix} T_{xm} \\ T_{ym} \end{bmatrix} Exp(-j\vec{k}_{3m}.\hat{z}(z - \ ))f_m(x, y).$$

For the grating region , we have the follow ıg coupled wave expansion for the electric and magnetic fields

$$\begin{Bmatrix} E_{2x} \\ E_{2y} \\ E_{2z} \end{Bmatrix} = \sum_{m=-\infty}^{\infty} \begin{Bmatrix} S_{xm}(z) \\ S_{ym}(z) \\ S_{zm}(z) \end{Bmatrix} Exp(-jk_{z0}z)f_m(x, \ .$$

$$\begin{Bmatrix} H_{2x} \\ H_{2y} \\ H_{2z} \end{Bmatrix} = \sum_{m=-\infty}^{\infty} \begin{Bmatrix} U_{xm}(z) \\ U_{ym}(z) \\ U_{zm}(z) \end{Bmatrix} Exp(-jk_{z0}z)f_m(x, \ ).$$

The refractive index is periodic in x ; d y and its Fourier Series expansion is

$$\epsilon(x, y) = \epsilon_0 \sum_p \epsilon_p \ Exp\left(j\frac{2\pi p}{d}x\right)$$

Applying ,Maxwell's equations

$$\vec{\nabla} X \vec{E}_2 = -j\omega\mu_0 \vec{H}_2$$

$$\vec{\nabla} X \vec{H}_2 = j\omega\epsilon(x, y, z)\vec{E}_2$$

Expanding the curl equations we have,

$$\frac{\partial E_{2z}}{\partial y} - \frac{\partial E_{2y}}{\partial z} = -j\omega\mu_0 H_{2x}$$
$$\frac{\partial E_{2x}}{\partial z} - \frac{\partial E_{2z}}{\partial x} = -j\omega\mu_0 H_{2y}$$
$$\frac{\partial E_{2y}}{\partial x} - \frac{\partial E_{2x}}{\partial y} = -j\omega\mu_0 H_{2z}$$

Substituting for the field expression

$$\sum_m \left[(-j\vec{k}_1.\hat{y}) S_{zm}(z) - (-jk_{z0}) S_{ym}(z) - \frac{\partial}{\partial z} S_{ym}(z) \right] Exp(-jk_{z0}z) f_m(x,y)$$
$$= -j\omega\mu_0 \sum_m U_{xm}(z) Exp(-jk_{z0}z) f_m(x,y)$$

$$\sum_m \left[(-j\vec{k}_{1m}.\hat{x}) S_{zm}(z) - (-jk_{z0}) S_{xm}(z) - \frac{\partial}{\partial z} S_{xm}(z) \right] Exp(-jk_{z0}z) f_m(x,y)$$
$$= j\omega\mu_0 \sum_m U_{ym}(z) Exp(-jk_{z0}z) f_m(x,y)$$

$$\sum_m \left[(-j\vec{k}_{1m}.\hat{x}) S_{ym}(z) - (-j\vec{k}_1.\hat{y}) S_{xm}(z) \right] Exp(-jk_{z0}z) f_m(x,y)$$
$$= -j\omega\mu_0 \sum_m U_{zm}(z) Exp(-jk_{z0}z) f_m(x,y)$$

And for the magnetic curl equation

$$\sum_{m,n} \left[(-j\vec{k}_1.\hat{y}) U_{zm}(z) - (-jk_{z0}) U_{ym}(z) - \frac{\partial}{\partial z} U_{ym}(z) \right] Exp(-jk_{z0}z) f_m(x,y)$$
$$= j\omega\epsilon_0 \sum_m \sum_p S_{xp}(z) \epsilon_{m-p} Exp(-jk_{z0}z) f_m(x,y)$$

$$\sum_m \left[(-j\vec{k}_{1m}.\hat{x}) U_{zm}(z) - (-jk_{z0}) U_{xm}(z) - \frac{\partial}{\partial z} U_{xm}(z) \right] Exp(-jk_{z0}z) f_m(x,y)$$

$$= -j\omega\epsilon_0 \sum_m \sum_p S_{yp}(z)\epsilon_{m-p} \exp(-jk_{z0}z) f_m(x,y)$$

$$\sum_m \left[(-j\vec{k}_{1m}.\hat{x}) U_{ym}(z) - (-j\vec{k}_1.\hat{y}) U_{xm}(z) \right] \exp(-jk_{z0}z) f_m(x,y)$$

$$= j\omega\epsilon_0 \sum_m \sum_p S_{zp}(z)\epsilon_{m-p} \exp(-jk_{z0}z) f_m(x,y)$$

$$S_{zm}(z) = \frac{1}{\omega\epsilon_0} \sum_p \left[(-\vec{k}_{1p}.\hat{x}) U_{yp}(z) + (\vec{k}_{1p}.\hat{y}) U_{xp}(z) \right] \bar{\epsilon}_{m-p}$$

$$\frac{1}{\epsilon(x,y,z)} = \frac{1}{\epsilon_0} \sum_p \bar{\epsilon}_p \exp\left(j\frac{2\pi p}{d} \right)$$

$$\frac{\partial}{\partial z} S_{xm}(z) = \left[(-j\vec{k}_{1m}.\hat{x}) S_{zm}(z) - (-jk_{z0}) S_{xm}(z) \right] - j\omega\mu_0 U_{ym}(z)$$

$$\frac{\partial}{\partial z} S_{ym}(z) = \left[(-j\vec{k}_1.\hat{y}) S_{zm}(z) - (-jk_{z0}) S_{ym}(z) \right] + j\omega\mu_0 U_{xm}(z)$$

$$U_{zm}(z) = \frac{1}{\omega\mu_0} \left[(\vec{k}_{1m}.\hat{x}) S_{ym}(z) - (\vec{k}_1.\hat{y}) S_{xm}(z) \right]$$

$$\frac{\partial}{\partial z} U_{xm}(z) = \left[(-j\vec{k}_{1m}.\hat{x}) U_{zm}(z) - (-jk_{z0}) U_{xm}(z) \right] + j\omega\epsilon_0 \sum_p S_{yp}(z)\epsilon_{m-p}$$

$$\frac{\partial}{\partial z} U_{ym}(z) = \left[(-j\vec{k}_{1m}.\hat{y}) U_{zm}(z) - (-jk_{z0}) U_{ym}(z) \right] - j\omega\epsilon_0 \sum_p S_{xp}(z)\epsilon_{m-p}$$

Eliminating the components of tl electric and magnetic fields normal to tl boundary (i.e z components) results in a et of four first–order coupled wave equation:

$$\frac{\partial}{\partial z}S_{xm}(z) = \frac{-j\vec{k}_{1m}.\hat{x}}{\omega\epsilon_0}\sum_p\left[(-\vec{k}_{1p}.\hat{x})U_{yp}(z) + (\vec{k}_1.\hat{y})U_{xp}(z)\right]\bar{\epsilon}_{m-p}$$
$$- (-jk_{z0})S_{xm}(z) - j\omega\mu_0 l_{ym}(z)$$

$$\frac{\partial}{\partial z}S_{ym}(z) = \frac{-j\vec{k}_1.\hat{y}}{\omega\epsilon_0}\sum_p\left[(-\vec{k}_{1p}.\hat{x})U_{yp}(z) - (\vec{k}_{1p}.\hat{y})U_{xp}(z)\right]\bar{\epsilon}_{m-p}$$
$$- (-jk_{z0})S_{ym}(z) + j\omega\mu_0 l_{xm}(z)$$

$$\frac{\partial}{\partial z}U_{xm}(z) = \frac{-j\vec{k}_{1m}.\hat{x}}{\omega\mu_0}\left[(\vec{k}_{1m}.\hat{x})S_{ym}(z) - \vec{k}_{1m}.\hat{y})S_{xm}(z)\right]$$
$$- (-jk_{z0})U_{xm}(z) + j\omega\epsilon_0\sum_p S_{yp}(z)\epsilon_{m-p}$$

$$\frac{\partial}{\partial z}U_{ym}(z) = \frac{-j\vec{k}_1.\hat{y}}{\omega\mu_0}\left[(\vec{k}_{1m}.\hat{x})S_{ym}(z) - 1.\hat{y})S_{xm}(z)\right]$$
$$- (-jk_{z0})U_{ym}(z) - j\omega\epsilon_0\sum_p S_{xp}(z)\epsilon_{m-p}$$

In order to solve for the scattered fie s, the infinite representation for the electric a magnetic fields has to be first truncated, o that the indices now run from –N to N. Tl truncation implies that the electric or ma netic field is now represented by a finite set fourier coefficients. Total number of un nowns to be solved for is 2*(2N + 1). The coefficeints are mapped into a single vect in the following manner. Define a vector $\bar{R}$ length 2*(2N+1) ,

$$\bar{R}[i] = \begin{cases} R_{x(i - [N+1])}, & \forall i \leq 2N+1, \\ R_{y(i-3*[N+1])}, & \forall i \text{ s.t. } 2N+1 < i \leq 2*[2N+1]. \end{cases}$$

The vector $\bar{T}$, $\bar{S}$, $\bar{U}$ are similarly defir d. To make things easier, define a mapping ξ

$$\xi[i] = \begin{cases} (i - [N+1]), & \forall i \leq 2N+1, \\ (i - 3*[N+1]), & \forall i \text{ s.t } 2N+1 < i \leq 2*[2N+1]. \end{cases}$$

The four coupled wave equations n y be written in matrix form as

$$\begin{bmatrix} S_{xi} \\ S_{yi} \\ \frac{\omega\mu_0}{k} U_{xi} \\ \frac{\omega\mu_0}{k} U_{yi} \end{bmatrix} = \begin{bmatrix} A & \Phi & C & D \\ \Phi & F & G & H \\ Q & J & K & \Phi \\ M & N & \Phi & P \end{bmatrix} \begin{bmatrix} S_{xi} \\ S_{yi} \\ \frac{\omega\mu_0}{k} U \\ \frac{\omega\mu_0}{k} U \end{bmatrix}$$

where Φ is the null matrix and A through (are matrices defined as follows. The dimensic of all the sub matrices are MN x MN. Th sub–matrix elements are defined as follows $A = F = K = P = jk_{z0}I$ where I the identity matrix.

$$C[p,l] = \frac{-j\vec{k}_{1\xi(p)}.\hat{x}}{k} \vec{k}_1.\hat{y}\bar{\epsilon}_{\xi(p)\,-\,\xi(l)}$$

$$D[p,l] = \frac{j\vec{k}_{1\xi(p)}.\hat{x}}{k} \bar{k}_{1\xi(l)}.\hat{x}\bar{c}_{\xi(p)\,-\,\xi(l)} - jk\bar{c}$$

$$G[p,l] = \frac{-j\vec{k}_1.\hat{y}}{k} \vec{k}_1.\hat{y}\bar{\epsilon}_{\xi(p)\,-\,\xi(l)}$$

$$H[p,l] = \frac{j\vec{k}_1.\hat{y}}{k} \bar{k}_{1\xi(l)}.\hat{x}\bar{\epsilon}_{\xi(p)\,+\,\xi(l)} - jk\delta_{pl}$$

$$Q[p,l] = \frac{j\vec{k}_{1\xi(p)}.\hat{x}}{k} \bar{k}_1.\hat{y}\delta_{pl}$$

$$J[p,l] = \frac{-j\vec{k}_{1\xi(p)}.\hat{x}}{k} \bar{k}_{1\xi(p)}.\hat{x}\delta_{pl} + jk\epsilon_{\xi(p)\,-\,\xi(l)}$$

$$M[p,l] = \frac{j\vec{k}_1.\hat{y}}{k} \bar{k}_{1(p)}.\hat{y}\delta_{pl} - jk\epsilon_{\xi(p)\,-\,\xi(}$$

$$N[p,l] = \frac{-j\vec{k}_1.\hat{y}}{k} \bar{k}_{1\xi(p)}.\hat{x}\delta_{pl}$$

Note: let $\xi(p)$ = (m) and $\xi(l)$ = (i) , then $\xi(p) - \xi(l)$ = (m–i)

The above matrix equation is solve: using the state–variable method as before. The next step is to match the electric and m: gnetic fields at the boundaries and this is very similar to the line grating case.

Coupled Wave Solution

$$S_{xi}(z) = \sum_{m=1}^{MM2} C_m w_{1,im} Exp[(\lambda_m - jk_z) z] \qquad \text{for } 1 \le i \le M2$$

$$S_{yi}(z) = \sum_{m=1}^{MM2} C_m w_{2,im} Exp[(\lambda_m - jk_z) z] \qquad \text{for } 1 \le i \le M2$$

$$\omega\mu_0 U_{xi}(z) = \sum_{m=1}^{MM2} C_m w_{3,im} Exp[(\lambda_m - jk_{z0}) z] \qquad \text{for } 1 \le i \le M2$$

$$\omega\mu_0 U_{yi}(z) = \sum_{m=1}^{MM2} C_m w_{4,im} Exp[(\lambda_m - jk_{z0}) z] \qquad \text{for } 1 \le i \le M2$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}$$

M2 = 2N + 1

MM2 = 4*M2

W is the eigenvector matrix. w_1 is a sub m trix that has M2 rows and MM2 columns. λ_m is the mth eigenvalue. There are MM2 total igenvalues.

Boundary Conditions Applications

The boundary conditions that have b applieed are the following.

i) Continuity of Tangential Elect c and Tangential Magnetic fields at the z=0 boundary.

ii) Continuity of Tangential Electri and Tangential Magnetic fields at the boundary between the grating and the underlying la er at the z = d boundary.

z = 0 Boundary

Now we need to calculate the tangential magnetic fields in region 1

$$\frac{\partial E_{1z}}{\partial y} - \frac{\partial E_{1y}}{\partial z} = -j\omega\mu_0 H_{1x}$$

$$\frac{\partial E_{1x}}{\partial z} - \frac{\partial E_{1z}}{\partial x} = -j\omega\mu_0 H_{1y}$$

$$\frac{\partial E_{1x}}{\partial x} + \frac{\partial E_{1y}}{\partial y} + \frac{\partial E_{1z}}{\partial z} = 0$$

$$\frac{\partial E_{1z}}{\partial z} = -\sum_{i=1}^{M2}\left((-j\vec{k}_{1i}.\hat{x})\alpha\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + (-j\vec{k}_{1i}.\hat{x})R_{xi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_{mn}(x,y)\right)$$

$$\sum_{i=1}^{M2}\left((-j\vec{k}_1.\hat{y})\beta\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + (-j\vec{k}_1.\hat{y})R_{yi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_{mn}(x,y)\right)$$

$$H_{1x} = \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})(-j\vec{k}_1.\hat{y})}{-j\vec{k}_{1i}.\hat{z}}\left(-\alpha\delta Exp(-j\vec{k}_1.\vec{r}) + R_{xi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_{mn}(x,y)\right)\right]$$

$$+ \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_1.\hat{y})(-j\vec{k}_1.\hat{y})}{-j\vec{k}_{1i}.\hat{z}}\left(-\beta\delta_{i0} xp(-j\vec{k}_1.\vec{r}) + R_{yi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_{mn}(x,y)\right)\right]$$

$$+ \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left((-j\vec{k}_1.\hat{z})\beta\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + (-j\vec{k}_{1i}.\hat{z})R_{yi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_{mn}(x,y)\right)$$

At z = 0, this becomes

$$H_{1x} = \frac{-1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{1i}.\hat{z}}\left(-\alpha\delta_{i0} + \quad z\right) + \frac{(\vec{k}_1.\hat{y})^2}{\vec{k}_{1i}.\hat{z}}(-\beta\delta_{i0} + R_{yi})f_{mn}(x,y)\right.$$

$$\left. - \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left((\vec{k}_{1i}.\hat{z})(R_{yi} - \beta\delta_{i0})f_{mn}(x, \quad)\right)\right.$$

$$H_{1x} = \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{1i}.\hat{z}}(\alpha\delta_{i0} - R_x + \left[\frac{(\vec{k}_1.\hat{y})^2}{\vec{k}_{1i}.\hat{z}} + \vec{k}_{1i}.\hat{z}\right](\beta\delta_{i0} - R_{yi})\right]$$

$$H_{1y} = \frac{-1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})^2}{-j\vec{k}_{1i}.\hat{z}}\left(-\alpha\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + R_{xi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_i(x,y)\right)\right]$$

$$- \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})(-j\vec{k}_1.\hat{y})}{-j\vec{k}_{1i}.\hat{z}}\left(-\beta\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + R_{yi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_i(x,y)\right)\right]$$

$$- \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left((-j\vec{k}_1.\hat{z})\alpha\delta_{i0}Exp(-j\vec{k}_1.\vec{r}) + (-j\vec{k}_{1i}.\hat{z})R_{xi}Exp(-j\vec{k}_{1i}.\hat{z}z)f_i(x,y)\right)$$

At the z = 0 boundary

$$H_{1y} = -\frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})^2}{\vec{k}_{1i}.\hat{z}}\left(-\alpha\delta_{i0} + R_{xi}\right) + \frac{(\vec{k}_1.\hat{y})(\vec{k}_{1i}.\hat{x})}{\vec{k}_{1i}.\hat{z}}(-\beta_{i0} + R_{yi}\right]f_{mn}(x,y)$$

$$+ \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left((\vec{k}_{1i}.\hat{z})\left(R_{xi} - \alpha\delta_{i0}\right)f_{mn}(x,y)\right)$$

$$H_{1y} = \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{1i}.\hat{z}}\left(-\beta\delta_{i0} + R_{yi}\right) + \left[\frac{(\vec{k}_{1i}.\hat{x})^2}{\vec{k}_{1i}.\hat{z}} + \vec{k}_{1i}.\hat{z}\right]\left(R_{xi} - \alpha\delta_{i0}\right)\right]$$

The boundary conditions at z = 0 can then be written as

$$\alpha\delta_{i0} + R_{xi} = \sum_{m=1}^{MM2} C_m w_{1,im}$$

$$\beta\delta_{i0} + R_{yi} = \sum_{m=1}^{MM2} C_m w_{2,im}$$

$$\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{1i}.\hat{z}}\left(\alpha\delta_{i0} - R_{xi}\right) + \left[\frac{(\vec{k}_1.\hat{y})^2}{\vec{k}_{1i}.\hat{z}} + \vec{k}_{1i}.\hat{z}\right]\left(\beta\delta_{i0} - R_{yi}\right)\right.$$

$$= \sum_{m=1}^{MM2} C_m w_{3,im}$$

$$\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{1i}.\hat{z}}\left(R_{yi} - \beta\delta_{i0}\right) + \left[\frac{(\vec{k}_{1i}.\hat{x})^2}{\vec{k}_{1i}.\hat{z}} - \vec{k}_{1i}.\hat{z}\right]\left(R_{xi} - \alpha\delta_{i0}\right)\right.$$

$$= \sum_{m=1}^{MM2} C_m w_{4,im}$$

z = d Boundary

Now we nced to calculate the tangential magnetic fields in region 3

$$\frac{\partial E_{3z}}{\partial y} - \frac{\partial E_{3y}}{\partial z} = -j\omega\mu_0 H_{3x}$$

$$\frac{\partial E_{3x}}{\partial z} - \frac{\partial E_{3z}}{\partial x} = -j\omega\mu_0 H_{3y}$$

$$\frac{\partial E_{3x}}{\partial x} + \frac{\partial E_{3y}}{\partial y} + \frac{\partial E_{3z}}{\partial z} = 0$$

$$\frac{\partial E_{3z}}{\partial z} = -\sum_{i=1}^{M2}\left((-j\vec{k}_{1i}.\hat{x})T_{xi} - (-j\vec{k}_{1}.\hat{y})T_{yi}\right)Exp(-j\vec{k}_{3i}.\hat{z}(z-d))f_{mn}(x,y)$$

$$H_{3x} = \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})(-j\vec{k}_{1}.\hat{y})}{-j\vec{k}_{3i}.\hat{z}}\left(T_{xi}Exp(-j\vec{k}_{3i}.\hat{z}(z-d))f_{mn}(x,y)\right)\right]$$

$$+\frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1}.\hat{y})(-j\vec{k}_{1}.\hat{y})}{-j\vec{k}_{3i}.\hat{z}}\left(T_{yi}Exp(-j\vec{k}_{3i}.\hat{z}(z-d))f_{mn}(x,y)\right)\right]$$

$$+\frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left((-j\vec{k}_{3i}.\hat{z})T_{yi}Exp(-j\vec{k}_{3i}.\hat{z}(z-d))f_i(x,y)\right)$$

At z = d, this becomes

$$H_{3x} = \frac{-1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_{1}.\hat{y})}{\vec{k}_{3i}.\hat{z}}T_{xi} + \frac{(\vec{k}_{1}.\hat{y})^2}{\vec{k}_{3i}.\hat{z}} - T_{yi}\right]f_i(x,y)$$

$$-\frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left((\vec{k}_{3i}.\hat{z})T_{yi}f_i(x,y)\right)$$

$$H_{3x} = \frac{-1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_{1}.\hat{y})}{\vec{k}_{3i}.\hat{z}}T_{xi} + \left[\frac{(\vec{k}_{1}.\hat{y})^2}{\vec{k}_{3i}.\hat{z}} + \vec{k}_{3i}.\hat{z}\right]T_{yi}\right]f_i(x,y)$$

$$H_{3y} = \frac{-1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})^2}{-j\vec{k}_{3i}.\hat{z}}T_{xi}Exp(-j\vec{k}_{3i}.\hat{z}(z - d))f_i(x,y)\right]$$

$$- \frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(-j\vec{k}_{1i}.\hat{x})(-j\vec{k}_1.\hat{y})}{-j\vec{k}_{3i}.\hat{z}}T_{yi}Ex\left(-j\vec{k}_{3i}.\hat{z}(z - d)\right)f_i(x,y)\right]$$

$$-\frac{1}{j\omega\mu_0}\sum_{i=1}^{M2}(-j\vec{k}_{3i}.\hat{z})T_{xi}Exp(-j\vec{k}_{3i}.\hat{z}(z - d))f_i(x,y)$$

At z = d boundary

$$H_{3y} = \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})^2}{\vec{k}_{3i}.\hat{z}}T_{xi} + \frac{(\vec{k}_1.\hat{y})(\vec{k}_{1i}.\hat{x})}{\vec{k}_{3i}.\hat{z}}T_{yi}\right]f_i(x,y)$$

$$+ \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}(\vec{k}_{3i}.\hat{z})T_{xi}f_i(x,y)$$

$$H_{3y} = \frac{1}{\omega\mu_0}\sum_{i=1}^{M2}\left[\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{3i}.\hat{z}}T_{yi} + \left[\frac{(\vec{k}_1.\hat{\epsilon})^2}{\vec{k}_{3i}.\hat{z}} + \vec{k}_{3i}.\hat{z}\right]T_{xi}\right]f_i(x,y)$$

The boundary conditions can then be written as

$$T_{xi} = \sum_{m=1}^{MM2} C_m w_{1,m}Exp[(\lambda_m - jk_{z0})d]$$

$$T_{yi} = \sum_{m=1}^{MM2} C_m w_{2,m}Exp[(\lambda_m - jk_{z0})d]$$

$$\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{3i}.\hat{z}}\left(- T_{xi}\right) - \left[\frac{(\vec{k}_1.\hat{y})^2}{\vec{k}_{3i}.\hat{z}} + \vec{k}_{3i}.\hat{z}\right]\left(T_{yi}\right)$$

$$= \sum_{m=1}^{MM2} C_m w_{3,m}Exp[(\lambda_m - jk_{z0})d]$$

$$\frac{(\vec{k}_{1i}.\hat{x})(\vec{k}_1.\hat{y})}{\vec{k}_{3i}.\hat{z}}T_{yi} + \left[\frac{(\vec{k}_{1i}.\hat{x})^2}{\vec{k}_{3i}.\hat{z}} + \vec{k}_{3i}.\hat{z}\right]T_{xi}$$

$$= \sum_{m=1}^{MM2} C_m w_{4,m}Exp[(\lambda_m - jk_{z0})d]$$

Distribution:

MS-0318 George Davidson, 9215
MS-0321 W. J. Camp, 9200
MS-0439 David Martinez, 9234
MS-0819 James Peery, 9231
MS-0820 Paul Yarrington, 9232
MS-0828 Elaine Gorham, 9209
MS-1074 Bruce Draper, 1306
MS-1109 Art Hale, 9224
MS-1109 Richard J. Pryor, 9202
MS-1110 David Greenberg, 9223
MS-1110 David Womble, 9222
MS-1111 Grant Heffelfinger, 9225
MS-1111 Sudip Dosanjh, 9221
MS-0441 Robert Leland, 9226

1 MS-0161 Patent and Licensing Office, 11500
1 MS-9018 Central Technical Files, 8940-2
5 MS-0899 Technical Library, 4916
2 MS-0619 Review and Approval Desk, 12690